

2672

The opinion in support of the decision being entered today was not written for publication and is not binding precedent of the Board.

Paper No. 29

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS  
AND INTERFERENCES

**RECEIVED**

OCT - 1 2003

DIRECTOR OFFICE  
TECHNOLOGY CENTER 2000

**MAILED**

**SEP 24 2003**

U.S. PATENT AND TRADEMARK OFFICE  
BOARD OF PATENT APPEALS  
AND INTERFERENCES

Ex parte ROBERT A. FABBIO,  
ANNE G. LEONARD,  
and CONRAD W. SCHNEIKER

Appeal No. 1994-1648  
Application 07/352,530<sup>1</sup>

ON BRIEF

Before BARRETT, FLEMING, and DIXON, Administrative Patent Judges.  
BARRETT, Administrative Patent Judge.

DECISION ON APPEAL

This is a decision on appeal under 35 U.S.C. § 134 from the final rejection of claims 1-27.

We reverse.

<sup>1</sup> Application for patent filed May 15, 1989, entitled "Object Database-Driven Interactive Shell for a Data Processing System."

Appeal No. 1994-1648  
Application 07/352,530

#### BACKGROUND

This appeal was remanded to the examiner (Paper No. 22, May 20, 1997) for evaluation of the rejection of claim 27 under 35 U.S.C. § 101 in view of the Examination Guidelines for Claims Reciting a Means or Step Plus Function Limitation In Accordance With 35 U.S.C. § 112, 6th paragraph, 1162 Off. Gaz. Pat. & Trademark Office 59 (May 17, 1984) and the Examination Guidelines for Computer-Related Inventions, 1184 Off. Gaz. Pat. & Trademark Office 87 (March 26, 1996). The examiner withdrew several rejections of claim 27, but maintained the rejection of claims 1-27 under § 103 over Beck et al. (Paper No. 25, July 27, 1999). Appellants maintained their traverse of the remaining rejection of claims 1-27 over Beck et al. for the reasons stated in the brief (Paper No. 25½, August 30, 1999). The examiner acknowledged receipt of appellants' reply (Paper No. 26, November 15, 2002). A date for oral hearing was scheduled for August 21, 2003 (Paper No. 27, May 28, 2003). The oral hearing was waived (Paper No. 28, June 16, 2003).

The invention relates to an interface (claim 1), a method (claim 26), and a program stored on a computer compatible medium (claim 27) for presenting items for selection by a user of a data processing system. The invention is fairly summarized by appellants in the Summary of the Invention in the brief.

Claim 1 is reproduced below.

1. An interface having means for presenting items for selection by a user of a data processing system, and having means for executing the items which are selected, said interface comprising:

means for representing a plurality of interface objects in an object database; and

means for dynamically associating different ones of said interface objects with a plurality of logical frame presentations based upon data within each of said different ones of said interface objects.

The examiner relies on the following reference:

[illegible]

Claims 1-27 stand rejected under 35 U.S.C. § 103 as being unpatentable over Beck. The examiner refers (examiner's answer, page 5) to Beck, Fig. 5, items 27, 40, 41, and 53 for the "means for representing a plurality of . . . objects," "means for dynamically associating different ones," and "plurality of logical frame presentations based upon the data within each of said different ones." The examiner notes that Beck does not explicitly teach "interface objects," but concludes (examiner's answer, page 5): "[I]t would have been obvious . . . to use interface objects because the reference discloses graphical representations of objects for the purpose of monitoring program execution. Thus, the graphical representation of objects can be construed as interface objects."

We refer to the final rejection (Paper No. 8) and the examiner's answer (Paper No. 19) (pages referred to as "EA\_\_") for a statement of the examiner's rejection, and to the brief (Paper No. 17) (pages referred to as "Br\_\_") for a statement of appellants' arguments thereagainst.<sup>2</sup>

OPINION

Beck

Beck describes that higher level languages such as Basic, Fortran, and Pascal are "sequential" languages where programs consist of code generally listed in the order in which the functions represented by the code are to be performed by the computer (col. 1, lines 13-32). In contrast to sequentially organized software, "object-oriented" software is organized into "objects," each comprising a block of computer instructions describing various procedures ("methods") to be performed in response to "messages" sent to the object (col. 1, lines 50-54). Although object-oriented software makes simulation of systems of interrelated components more intuitive, the operation of an object-oriented program is often more difficult to understand

---

<sup>2</sup> Appellants filed a reply brief (Paper No. 20, July 21, 1993) directed to the new grounds of rejection of claim 27 under 35 U.S.C. § 101 and § 112, first and second paragraphs in the examiner's answer (EA11-21). The examiner filed a supplemental examiner's answer (Paper No. 21, October 27, 1993). Since these rejections of claim 27 have been withdrawn, we do not refer to the reply brief or the supplemental examiner's answer.

because the sequence of operations is usually not immediately apparent from a software listing as in the case of sequentially organized programs (col. 2, lines 32-38). Beck discloses a "diagramming debugger" which displays a graphical representation of the sequences of messages sent during operation of an object-oriented program (abstract). The operation is succinctly described in the abstract of Beck and is shown in Fig. 5.

#### Obviousness

The claims stand or fall together.

Appellants argue (Br7) that Beck fails to teach or suggest the limitation of "dynamically associating different ones of said interface objects with a plurality of logical frame presentations based upon data within each of said different ones of said interface objects." It is argued that the use of data within the object to dynamically associate objects within the frame presentation is a key aspect of the claimed invention (Br7). Objects can be added without the need of re-compilation since the claimed invention is data driven and not predefined in the programming code (Br8-9). Appellants argue that Beck's graphical representations are passive and merely depict a transmitting and receiving object (Br7). It is argued (Br8):

There is no suggestion or teaching in Beck of using data within the object itself to aid, assist, or be used in conjunction with the dynamic association of objects with the screen representation. If Beck has any dynamic association at all, it is the use of messages to invoke screen

representations of objects. These Beck messages are not contained within the objects to be displayed by a screen representation. Rather the messages are used to convey information between, and thus outside of, the objects.

The examiner parses the limitation into the terms and phrases "dynamically," "dynamically associating," "interface objects," and "based upon the data within each of said . . . interface objects," and concludes that each, as interpreted by one of ordinary skill in the art, is taught by Beck and thus the limitation as a whole is taught by Beck (EA22-24).

This is the first time that the examiner has made this claim interpretation, and we have no response by appellant. Nevertheless, we are not persuaded that the examiner is correct. Initially, although the program objects in Beck are not intended to be "interface objects," in the sense of menu, name, or dialog interface objects as in the disclosed invention, because the objects are displayed, we assume that they can be broadly termed "interface objects."

The examiner interprets "dynamically" as follows (EA22-23):

Taking the word "dynamically", it is clear that the meaning of "dynamically" is broad. It encompasses . . . the designation of an event that occurs during the execution of a computer program. Taking the last meaning, it is clear that all events described in the Beck patent occur during the execution of a computer program, and are therefore dynamic. Beck teaches in the abstract that his system "creates a graphical representation of the sequences of message sent during operation of an object-oriented program." Therefore, "dynamically", as reasonable [sic] interpreted by one of ordinary skill in the art, is taught by Beck.

Appeal No. 1994-1648  
Application 07/352,530

It is true that "dynamically" can mean "occurring during execution (run time), as opposed to being predetermined."

"Dynamic" is defined, IBM Dictionary of Computing (10th ed. Aug. 1993):

(1) In programming languages, pertaining to properties that can only be established during the execution of a program; for example, the length of a variable-length data object is dynamic; (2) Pertaining to an operation that occurs at the time it is needed rather than at a predetermined or fixed time; (3) Pertaining to events occurring at run time, or during processing; (4) Contrast with static.

However, the term "dynamically" modifies the rest of the limitation "associating different ones of said interface objects with a plurality of logical frame presentations based upon data within each of said different ones of said interface objects." Thus, the fact that something happens dynamically in Beck does not necessarily meet the rest of the claim limitation.

The examiner states (EA23):

Now taking the phrase "dynamically associating", this is taken to mean associating during the execution of a computer program; and the "graphical representation of the sequence of messages", of Beck, clearly teaches association. The "representation", i.e., the item which stands in place of "the sequence of messages", is clearly associated with "the sequence of messages." Therefore, "dynamically associating", as interpreted by one of ordinary skill in the art, is taught by Beck.

We do not agree that "graphical representation of the sequence of messages" meets the limitation of "dynamically associating different ones of said interface objects." The objects and interrelationships (associations) between objects in

Appeal No. 1994-1648  
Application 07/352,530

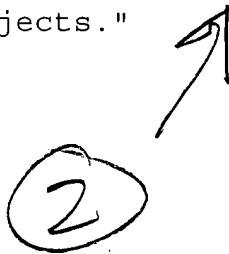
Beck are all statically predefined in the program; there is no dynamic association that occurs only at run time. That the objects are invoked and displayed during the program execution might suggest dynamically displaying predetermined object associations, but does not teach "dynamically associating different ones of said interface objects."

The examiner lastly states (EA23-24):

Finally, "based upon the data within each of said . . . interface objects" is taught by Beck in his recitation of "When one object transmits a message to another object . . . [the system] . . . displays representations of the transmitting and receiving objects on a computer screen" (see the abstract of Beck). The "data" upon which the associations are based are the "messages" which are transmitted among the objects. Therefore, "based upon the data", as interpreted by one of ordinary skill in the art, is taught by Beck.

For the reasons previously stated, we find that Beck does not teach "dynamically associating different ones of said interface objects," but at most suggests dynamically displaying predetermined object associations. Therefore, Beck does not teach "dynamically associating" "based upon data within each of . . . said interface objects."

2







Appeal No. 1994-1648  
Application 07/352,530

We conclude that the examiner has failed to establish a prima facie case of obviousness. The rejection of claims 1-27 is reversed.

REVERSED

Lee E. Barrett  
LEE E. BARRETT  
Administrative Patent Judge

  
MICHAEL R. FLEMING  
Administrative Patent Judge

  
JOSEPH L. DIXON  
Administrative Patent Judge

BOARD OF PATENT  
APPEALS  
AND  
INTERFERENCES

Appeal No. 1994-1648  
Application 07/352,530

WAYNE P. BAILEY  
IBM CORP., INTELLECTUAL PROPERTY LAW  
DEPT., 932/815, ZIP 4054  
11400 BURNET ROAD  
AUSTIN, TX 78758